Journal of King Saud University – Computer and Information Sciences (2015) 27, 181–192

King Saud University

Journal of King Saud University – Computer and Information Sciences

www.ksu.edu.sa
www.sciencedirect.com

Transformation rules for decomposing heterogeneous data into triples

Mrityunjay Singh *, S.K. Jain

National Institute of Technology, Kurukshetra 136119, India

Received 8 May 2013; revised 6 February 2014; accepted 13 March 2014
Available online 26 March 2015

KEYWORDS

Information integration;
Dataspace system;
Triple model;
Heterogeneity;
Transformation Rules Set;
Data modeling

Abstract In order to fulfill the vision of a dataspace system, it requires a flexible, powerful and versatile data model that is able to represent a highly heterogeneous mix of data such as databases, web pages, XML, deep web, and files. In literature, the triple model was found a suitable candidate for a dataspace system, and able to represent structured, semi-structured and unstructured data into a single model. A triple model is based on the decomposition theory, and represents variety of data into a collection of triples. In this paper, we have proposed a decomposition algorithm for expressing various heterogeneous data models into the triple model. This algorithm is based on the decomposition theory of the triple model. By applying the decomposition algorithm, we have proposed a set of transformation rules for the existing data models. The transformation rules have been categorized for structured, semi-structured, and unstructured data models. These rules are able to decompose most of the existing data models into the triple model. We have empirically verified the algorithm as well as the transformation rules on different data sets having different data models.

1. Introduction

In recent past, the attention has been made on the efficient management of the large volume of heterogeneous data distributed over several sites. Data integration is one way for managing such large collection of heterogeneous data but it has various shortcomings (Dong et al., 2009; El-Sappagh et al., 2011; Lenzerini, 2002). Recently , the dataspace approach has emerged as a new way of data integration which integrates the heterogeneous data in "pay-as-you-go" manner (Halevy et al., 2006; Franklin, 2009). This approach provides an incremental improvement over the existing data management systems for managing and querying the heterogeneous data in a uniform manner (Hedeler et al., 2009; Mirza et al., 2010). A dataspace is defined as a set of participants and a set of relationships among them. A participant may be any data source which contains data and may vary from structured to unstructured (Franklin et al., 2005; Singh and Jain, 2011). The examples of a dataspace system include Personal Information Management (PIM) (Dittrich et al., 2006; Dittrich et al., 2007), Scientific Data Management (Dessì and

* Corresponding author. Tel.: +91 8295594224.
E-mail addresses: mrityunjay.cse045@gmail.com (M. Singh), skj_nith@yahoo.com (S.K. Jain).
Peer review under responsibility of King Saud University.

Production and hosting by Elsevier

http://dx.doi.org/10.1016/j.jksuci.2014.03.017

Pes, 2009; Elsayed and Brezany, 2010), management of structured data on web such as Linked Data (Bizer et al., 2009; Ngomo, 2012; Van Hage et al., 2012).

The development of a dataspace system requires a simple and flexible data model for uniform representation of the heterogeneous data in a data-space. Previously, Halevy et al. have argued that a semi-structured graph based model is more suitable for dataspace systems (Halevy et al., 2006). Zhong et al. have advocated the use of Resource Description Framework (RDF) (Zhong et al., 2008) and proposed the *triple model* based on the RDF data model. A triple model is a simple and flexible data model based on the decomposition theory, which represents the heterogeneous data in data-space without losing their semantics. This model decomposes a large data unit into a set of smaller data units, and encapsulates each data unit into a triple.

In order to express various data models in triple model, and to avoid the uncertainty in data at various levels, a set of translation rules is required. In this work, we have employed the novel decomposition theory of triple model and proposed an algorithm which decomposes a data model into a collection of triples. Our algorithm works in two phases: *phase-1,* identifying all data item classes belonging to the input data model, and *phase-2,* decomposing each class to their respective components and encapsulating each component into a set of triples. Based on the decomposition algorithm, we have proposed a set of transformation rules for the structured, semi-structured and unstructured data models.

Previously, Zhong et al. present a set of decomposition rules w.r.t. a few data models (Zhong et al., 2008), whereas our work comprises of presenting a large set of transformation rules and a decomposition algorithm to apply on them. The proposed Transformation Rules Sets (TRSs) are exhaustive, and cover a broad range of data models in practical use. Therefore, these rules sets form a good base for implementation. One can extend these TRSs as well as the decomposition algorithm for other data models by identifying their respective classes and properties. We have applied our TRSs on various kinds of existing data models such as object relational, XML, iDM data model.

The rest of the paper is organized as follows: Section 2 presents the basic idea of the triple model. TRSs for various kinds of data models are presented in Section 3. The comparison and discussion of work are presented in Section 4 and 5 respectively. We have concluded our work in Section 6.

2. Triple model

A triple model is a graph based data model in which the smallest modeling unit is a triple. A *triple (T)* has three tuples (S, P, O), where S is a subject component, P is a predicate component, and O is an object component. Subject component(S) is a unique identifier of a data item, which is an integer type. Predicate component(P) has a 2-tuples (l, d), where l is a finite string that represents the label, and d is also a finite string which represents the data type. Object component(O) stores the actual data as an byte array.

A *data item (π)* is a unit populated in a dataspace which constitutes data such as a real world entity, a relation, a tuple, an xml element, a database, a file/folder, a web page. Before populating a data item in a dataspace, it must be decomposed into a collection of triples. For example, before populating the employee data item (e_1) in a dataspace, it must be decomposed into a set of triples as {(e_1, (*emp_name*, string), "R. Kumar"), (e_1, (*date_of_birth*, date), "17 /11/1983"), (e_1, (*date_of_joining*, date), "15/07/2009"), (e_1, (*organization*, string), "NIT"), (e_1, (*department*, string), "Computer engineering department"), and (e_1, (*salary*, currency), Rs 41,543/-)} as shown in Fig. 2.

A *data item class $C(\pi)$* is the predefined class for a data item. The set of data items having common properties are grouped into a data item class, e.g., files, folders, relations, XML elements, objects, web pages, an abstract entity like person. Every data item in a dataspace must belong to a predefined data item class otherwise we define a new class for this data item, e.g., a resource view class for a resource view data item in iDM model (Dittrich and Salles, 2006).

A *triple graph(G)* is a logical graph which is constructed among different triples populated in a dataspace. The triple graph (G) is defined as $G = (N, E, L)$, where N is a set of nodes. The internal nodes represent a data item with their identification, the leaf nodes represent the literal values which contain data. E is a set of edges. As shown in Fig. 1, an edge represents a relationship between either two data items (*i.e., association edge*) or a data item and its value (*i.e., attribute edge*) w.r.t property P. The association edge is represented as $< dataitem, association, dataitem >$, and the attribute edge is represented as $< dataitem, property, value >$. L is a set of labels on an edge with attribute or association name. Fig. 2 illustrates an example of triple graph in which the internal nodes are represented by an oval, and leaf nodes are represented by a dotted oval, a label on edge represents the predicate component of triple, and the direction of arrow is from subject to object of a triple.

A *Transformation Rule (TR)* maps a data model into the triple model without losing the semantics of data. The TRs for a data model depend on its respective properties. The collections of TRs related to a single data item class are grouped into the *Transformation Rules Set (TRS)*.

A *wrapper* is a program which extracts the desired data from its respective data sources, and transforms them into a collection of triples. A wrapper has two modules: a data extractor module and a data translator module. The data extractor module extracts the desired data from its respective data sources whereas the data translator module is based on TRSs, and translates the extracted data into a collection of triples. We have implemented a set of automatic/semi-automatic wrappers for the verification of the TRSs w.r.t. few data models such as structured data models (e.g., MySQL, PostgresSQL databases etc.), semi-structured data models (e.g., XML data, file system data, bibliographic data, LATEX data etc.), and unstructured data model (e.g., content of a text file, e-mails, web data, power point presentation etc.) (Singh and Jain, 2013). The set of automatic/semi-automatic wrappers can also be implemented for other existing data models based on the proposed TRSs. In the following section, we will explain the TRSs for the structured, semi-structured and unstructured data models .